COMAL TODAY 25

AMIGA
COMAL
SHIPS!

If the label says
Last Issue 25
You must renew now.
Use the order form inside.
No other notice is sent.

Bulk Rate
U.S.Postage
Paid
Madison, WI
Permit 2981

COMAL Today
5501 Groveland Ter
Madison, WI 53716

The Story of COMAL told by Borge Christensen

Are you going to be moving around a lot during your talk, or will you be fairly stationary?

What would you prefer?

Well, I don't mind.

I promise you that I will no do Danish Folk Dancing.

Oh.

That would be a disaster. We would probably be thrown out of NATO afterwards.

Anytime you are ready.

I'm ready now. Well, I'm going to simply talk. Well, I'd like to thank you all for coming. First, excuse me if sometimes it happens that I misspeak the words. It is a foreign language to me and my stock of words could all of a sudden disappear. You'll hear me searching for a word. Please bear with my language.

I have been so surprised to see the enthusiasm in America about this language of ours. What I'm going to talk about tonight might help you understand why I am surprised because this was from the start a very humble project, in a very small town, in a very small country, where it usually is quite cold, which might be one of the reasons why we are sitting so much indoors at our computers of course.

From the start this was a very humble project, in a very small town, in a very small country, where it usually is quite cold.

It started, in fact, in ... well, I might add one or two words about myself. I am hired by her Majesty, the Queen of Denmark, to teach Mathematics at a small college near the German border. When I say that I am hired by her Majesty, the Queen, it is actually true. I have a contract with her signature on it. Now you know how small our country really is. Her majesty does not go out each night and lock the door to the kingdom, as it says in Hans Andersons fairy tales. But, she can sign contracts, personally, with some professors, at some colleges, in some positions. So, she expects me to teach Mathematics, and that's what I'm doing.

Besides that I have to give courses in Computer Studies, specifically with applications of a mathematical kind. And that was how it all came about. In 1972 we got a Data General NOVA 1200 mini-computer, which some of you might know or have heard rumors about. We were supposed to start this new subject, Computer Science. Since I had long time ago taken a course at the University (only three weeks, but they didn't know that), I was expected to take care of this new thing, and try to teach the students how to use it.

The language that came with it was called Extended BASIC. One of the first things that we experienced was that it was so crowded with bugs that even the cleaning person began to complain about it. It was real bad. It broke down ... it was a so called core share system - no disk. It was only paper tape, all of it. Paper tape and teletypes. And it broke down regularly every four hours, so we had to re-install the programs by reading the paper tapes at a speed of 10 characters per seconds. It was quite a job. I think even this one would seem fast compared to that [referring to a Commodore Disk Drive]. This is, as you know, the worlds fastest tape station.

Another thing that seemed to go wrong was the way the students performed. That was to put it mildly ... miserably. The programs, even small programs were very bad. They didn't understand much of it, and it happened again and again that they could do it much easier on their pocket calculators.

Also, for myself, I found it extremely difficult to mark their works, because it was hard to read, simply to read their programs. Twenty or thirty line programs were hard to find out what was going on. I had to draw diagrams very often, simply to find out what they had did, what methods they had applied, what approach they had used. Of course, it might have been me that was not good enough, but that is an uncomfortable idea, that you're not good enough, so after awhile, I went to a nearby university, I knew their Computer Science section was a very good one ... in fact I went there because we had to set up some exam papers, some examination problems, and I didn't know what to do about this, because I could see disaster ahead, even if I only gave them simple problems to solve in this language, this so-called language, BASIC.

Here, in the afternoon, I met a man, his name was Benedict Lofstedt. When I came in and explained to him what this was all about, he said: *Well, this is not your fault, this is not the students fault, it is the language that you are using. It is in fact, not a computer language. It is an offense to the human mind.*

I am a mathematician, I'm not a computer scientist. And we were quite impressed by this new technology. It cost a whole lot of money and the local newspapers had written about it. So, I was surprised that he asked me to go back and read a new book by Niklaus Wirth called Systematic Programming. In that book he presented a language called Pascal. So we looked at it and I could see at once that this was what programming ought to be about.

Now, we couldn't use that Pascal directly. As it was, it was a very good language for advanced University students and professionals. But for my students, it would be impossible.

It is important to understand that there is a difference between the language itself, and the operating environment this language is working in.

There are one very good thing about BASIC ... it's interactive. You can type in small programs, have them run at once. You won't have to write a program using an editor, having the text stored, have the compiler load it, compile it ... going through all these phases of editing, compiling, and manipulating the operating system; that would be far too complicated for the students I had. So, we would have to maintain some of the user friendly environment there is in BASIC. I think that is the only good thing you can say about BASIC, it's the user environment. And it is important to understand that there is a difference between the language itself, and the operating environment this language is working in. We found out that when we tried to define the language.

Benedict Lofstedt, the computer scientist, and myself wrote letters to each other over a period of half a year, and tried to define another sort of language. We took what we had of BASIC, and added the structures of Pascal. So it is true, that we took the best from BASIC and the best from Pascal.

It would be very hard, I knew, to have it implemented. One thing is to define a programming language. We had finished the definition in fact by the beginning of 1974 - we had the first definition, of what later became COMAL 75. But to implement it is quite a different story. We went to one of the major companies and they asked me, rather cynically by the way, unfriendly, to come back again with $50,000; then they might consider the whole project. Otherwise we could go ahead with what we had. No understanding at all.

We went to one of the major companies and they asked me, rather cynically by the way, unfriendly, to come back again with $50,000.

Well, I then had what you would call a really good fortune. I got two very bright students. I did machine coding at the NOVA, which was in fact a 16 bit machine, and I told them how to code, machine code. In a few weeks they became much better at that than I was, than I ever became. They were young people, and had learned it very fast.

One day one of them came up, and said: *This definition of yours that Benedict and you have been doing. I think we can do that.*

I was very happy and surprised. Well try it.

They worked on it for about half a year, and eventually came up with the first implementation. We had a very close co-operation. Again and again we took discussions on how to implement the different details. I can still remember the day that we tried to put in long variable names. This was a problem then. There were no long variable names. Instead of saying number, name, we could only say N or N1. Still I think there are many BASICs that only allow short variable names.

We had a huge board where we wrote down all sorts of ideas. It was a great experience. It was quite fantastic, to be in that process with these two extremely clever young boys.

But we wanted them, wanted them right away in the first version to be at least 8 characters. We wanted to allow 8 characters. At least we would then be able to give variables sensible names. Such that the programs would be much more readable. This one of the first things we were after.

We found out somehow, that by putting these names in arrays, and using tokens to address them,

and thereby addressing their data fields, that we could do it that way. This sort of discussions were going on for half a year. We had a huge board where we wrote down all sorts of ideas. It was a great experience. It was quite fantastic, to be in that process with these two extremely clever young boys.

We poured coffee in him, and put him into a cold shower. Then he would work for 24 hours. It was just great.

I had a problem with one of them. He was a heavy drinker. Now and then he would go away, and we knew where he was. But only at a certain time when he had had enough could we pick him up. We could not persuade him to come back again until he had taken so and so much. He would then be very drunk. And we needed him, as he was our expert in debugging. Danish beer can be quite heavy, quite strong. We poured coffee in him, and put him into a cold shower. Then he would work for 24 hours. It was just great.

The first version of COMAL ... still we hadn't called it that, we had a Danish name for the project ... was running on the 5th of August, 1974. It was a five line program with a REPEAT ... UNTIL. And we had all the automatic indentation ... that was all done. We were working on the IF .. THEN .. ELSE with indentation. We pinned it on the board, and it was stolen a fortnight later, so I don't know where it is. This is quite a pity.

We had no disk! No diskette, no hard disk, no soft disk, or anything. We sent it out on paper tapes.

In February 1975, we had a version to send out. Now, you should imagine that we would send it out on a disk. We had no disk! No diskette, no hard disk, no soft disk, or anything. We sent it out on paper tapes this size.

I don't know if any of you have worked on a teletype.

Several people cried out Yes!

OK ... you know what it's like. Most would have a high speed reader, typically, 500 characters per second, which was quite respectable. But many of them would have to use a teletype 10 characters per second... it took one hour to load COMAL. So please don't complain over this. (God bless the poor fellow who then switched the whole thing off by some mistake).

Again and again we went around ... by the way, when we implemented, we only had paper tape. We had a fast paper tape punch and a fast paper tape reader, but we used miles on end of paper tape. Sometimes were walking around in paper tapes up to our knees. My job most of the time was to wind up ... I invented a very fine technique. I think I could even do it today if I had a chance ... wind it up very fast.

This was an experience of a lifetime, to be doing such a thing. It was absolutely fantastic.

One thing that might happen eventually, you probably know that ... I need to tell you that the whole implementation cost me $300. That was all we had, so we did it in our spare time ... we thought it was great fun. And it was. This was an experience of a lifetime, to be doing such a thing. It was absolutely fantastic.

When we were in the process of reading in a paper tape, it happened eventually that one stepped on it. And we could start all over again, because the paper tape broke. This was one of the deadly sins, I don't know how many are allowed.

We were fortunate, because most of the institutions that were of our type, (high schools, colleges, vocations schools, etc) had in fact Data General NOVAs, because one local Danish company was the distributor of that machine. We didn't have problems with such obscure machines such as Hewlett Packard or whatever that is. We had the NOVA all over. So they took it and started to use it. We were quite successful. I think it took half a year and they had stopped using the regular BASIC and turned over to COMAL.

Now, as it appeared, the major reason for doing that was the long variable names, believe it or not. Especially the students. The teachers were more hesitating, but the students took it very fast. They could now write NUMBER=5 or NAME$="Peter Sorenson".

And then the IF .. THEN .. ELSE. That was one that really was bought at once. I don't know if you are aware that doing branching, especially nested branching, using GOTO is in fact very very difficult. Therefore we had the nested IF ... ELSE ... ENDIF only up to a level of four at that time, but that appeared to be enough.

And then of course, the third facility, was the named procedures. We didn't have parameters at that time. You might call it simply named subroutines. Instead of saying GOSUB 5000, we could say EXECUTE PRINTOUT. And then you would have PROC PRINTOUT ... ENDPROC. That was all there at that time.

The first COMAL included:
- **Long Variable Names**
- **IF .. THEN .. ELSE**
- **Procedures & EXECUTE**

This was all much much easier. And it was faster. This was another thing we noted and which helped us. I don't know if you are aware that when BASIC tries to find line number 5000 starting at line number 10, it will go over all the line number to try to find it. In the first COMAL we did, we put up links, such that if there was an EXECUTE (that was the keyword we used) EXECUTE PRINTOUT in say line 50, and PROC PRINTOUT would appear in line 500, there was a pointer telling line 50 to jump at once (in machine code) to line 500. That of course was much much faster than having to go sequentially through all the line numbers. It was also one of the advantages that made COMAL popular very fast.

We were well aware that we should have some fancy English looking name, otherwise no one would have it.

This went on, and I gave it the name ... in fact, I can tell you how that happened ... the name COMAL. We had been discussing for a week what we should call it. We couldn't use the Danish name, we were well aware that we should have some fancy English looking name, otherwise no one would have it. In fact, I was on my bicycle from the college back home when it appeared to me, that we had been fooling around with the language ALGOL, which you probably know, ALGOrithmic Language, and all of a sudden it appeared that this should be COMMON Algorithmic Language. Shortened is COMAL. That was in fact how the name was made. That's all. It is a really country name.

That was, in fact, how the name was made. That's all. It is a really country name.

We continued to improve it a little ... as I said before, we then sent it out. I was not doing much more machine coding, because the students asked me to write the documentation. That's of course a very important thing. I also wrote a text book which later became Beginning COMAL. That took 3 or 4 years before it developed into the Danish edition of that book.

This went on until around 1978, when the first microcomputers appeared. We got a problem. The problem we had to face would be the following. Imagine the Danish schools had started to buy PET or APPLE. That meant that we were sent back to BASIC again. This was to me very frightening. It was so frightening in fact that if I had been forced to teach BASIC again, I would not have taught computers at all. That's true! I was completely determined, if they were to force me to teach BASIC again I would have said: **OK, you hired me to teach Mathematics, Algebra, Statistics, and Number Theory, and I'm good at that, so that's what I'm going to do. You can take your computers somewhere else.** I was absolutely determined to do that. But of course, it would be better if we could come up with a microcomputer with COMAL.

So in 1978 I defined a new version, where we put in the parameter mechanisms and the sort of things you know now, most of it at least. And we found another group in a college somewhere else that were willing to try to improve, enhance, expand COMAL into what we call COMAL 80 because it was not until 1980 when it was really finished, and it was running on a Z80 based computer, Zilog 80. So we had two reasons to call COMAL 80.

At the same time another Danish company designed and built a computer for schools. It was called the Comet. We are not always that modest in Denmark. It appeared to be a very good computer by the way. It is still used especially at technical schools

because it has one of those busses where you can add in extra electronics all the time to the buss. It is still very popular. That was the first computer to run a COMAL that you would recognize. If you saw programs written for that computer, you wouldn't be surprised at all. This would look very much like what you see today. What had to come would then be to make better environment, better editors, to put in extra facilities, such as graphics, sprites, and sound.

This was simply one of the most qualified critics that I ever had. It was very good.

But at that time, 1980-1981, I stopped doing much about it. It was then taken over by different groups, one of which is UniComal which is one of the most talented. Maybe I should add now about the way UniComal started. One of them Mogens Kjaer, wrote a letter to me, where he criticized my definition of COMAL 80. When I saw this letter, I knew that once that he would be one of the boys that could do something about it. This was simply one of the most qualified critics that I ever had. It was very good. I adjusted my COMAL according to what he advised me to do. And then I asked him if he might do ... he had intended to do an implementation for a computer that wouldn't have any chance of going anywhere. I advised him not to do that, but instead to go for the Commodore PET. It was obviously a very good computer.

A company could not sell a computer to a Danish school if it did not have COMAL.

At that time, you should know, by the way, a company could not sell a computer to a Danish school if it did not have COMAL. Hewlett Packard never sold their computer since it did not have COMAL. We had given it such a strong position. The teachers wouldn't have it. The teachers refused to use BASIC. That was a very strong situation. We hold in fact a very strong position without telling anybody. Simply by giving them a language that was apparently better.

Mogens Kjaer made this version for Commodore, and this is the version now known as 0.14. After he did that, the UniComal group was set up and they then went on. From then on I have not had much to do with it. These young fellows are doing a terrific good job. They added the packages, which is probably the most important extension to COMAL that was made after the latest definition. This was the package concept that you could have packages in machine code and call them.

From then on you know more or less what the story is all about. Still a new version now is being designed for the Z80 CP/M based computers for there is an English computer called the Amstrad, which is quite popular in Europe. They expect to sell around 600,000 of them. We have a version of COMAL for that now [this is the CP/M COMAL]. It is very much like the UniComal version. There are a few improvements, but not much.

What I'm interested in now ... I don't think we should expand COMAL much more. I feel perfectly well with the version for the Commodore 64. I have 17 of them for my students. We will probably have one or two PC's only because an institution like ours must have PC's. I probably can persuade someone to use them if we get them.

What I'm interested in now is to see a better user environment.

What I'm interested in now is to see a better user environment. It has improved very much with FIND and CHANGE. I would like to see full screen editors, without line numbers. There is a COMAL, Mytech [Alder] COMAL, with such a very good editor, but I'm sorry to say that the COMAL is not as good as it ought to be, but that may come. There is now an international standardization committee. We have a meeting in Scotland in September. We are going to discuss, among other things, whether we should get rid of line numbers, or at least make them redundant. You can get them if you like, but you can avoid them. You are aware I expect, that line numbers are not needed after we have left the teletype. You can with a screen editor, you can move the cursor wherever you like, and insert a line by manipulating some control code. Whereas with the teletype, line numbers were quite ingenious. You couldn't get back to the place between line 10 and 20, but you could print line 15 and have it inserted between line 10 and 20. This was in fact very well made considering that it was made for teletypes. But they are, I think, gone long ago.

If anyone had told me in 1974 that ten years later I would be standing in Los Angeles, I probably wouldn't have believed him.

This is more or less how it was. When we started it, we ... well if anyone had told me in 1974 that 10 years later I would be standing in Los Angeles, I probably wouldn't have believed him. Because we only wanted, as simple as that, was a tool that our students could use. Nothing pretentious or great project or anything. It appeared that our students could use it, it has developed over the years. It is therefore, as you may guess, very impressive, and makes me ... Well I don't know exactly how happy I really am, maybe I'll find out when I come back, maybe, that you really like it and are so enthusiastic about it. I think it is a good tool. It appears to be so. And I hope that many more people can get to know about it.

Well, you are welcome to ask questions and ... that's a lot of detail, but I may add that I used a slogan ... I hope you don't think I'm nasty ... I used a slogan that in the 70's I said that: **It might very well be that the good God has invented the computer. But there is no doubt that his black majesty then came up with BASIC shortly thereafter [laughter] ... in order to confuse peoples minds, because that's the way he prefers to operate. [applause].**

I would like to begin by taking a poll. Of all the COMAL programmer here, I would like to hear yays or nos from those who would like to see line numbers implemented for COMAL 2.0. All who would like to have line numbers ...

[interruption] What about how it is to be implemented? If you remove line numbers, how do you differentiate being in command mode and being in program mode.

That's right. Now, I think I said that it should be made redundant. But you would have to have two modes, an editing mode, and a runtime mode. In fact you can see it on the Mytech [Alder] COMAL, how it operates. There's no problems. I can demonstrate it after this meeting for whoever likes to see it. This is very easy to implement and use. But still you can use line numbers.

What would be the object of getting rid of the line numbers? What is the point? What is the big controversy of getting rid of them?

I would hate to admit that it is some academic idea. But I think it is. The idea is ... when I say full screen editor, I do not mean only the text you can see on the screen right now, but you should be able to scroll up and down. You can now in COMAL have the listing coming up, but you cannot take the cursor up and have the listing coming down again [note: this is now possible in IBM PC COMAL, AmigaCOMAL, and C128 cartridge COMAL]. So you should be able to look at the screen as simply a window, through which you pull a film either forward or backward.

But that is not the most important thing. I think I am best at giving examples because I am not a computer scientist, so this is not ... I am not one of those desktop academics that think these things out. I look at the students and try to find out what they need. And then we do it. COMAL is very pragmatic. We never took in new facilities and new structures until we felt that the students needed them. If they made a lot of mistakes, or if they had to do more work than necessary with details, then maybe an extra facility would make it much easier for them. That is in fact how the parameters were introduced. We discovered of course, that parameters would make it much easier for them, so we introduced them. But not until then.

That is why I'm a mathematician - I hate numbers. As a child, when I found out that you didn't have to use numbers, that you could use letters instead, that was it!

Now, imagine that you have a list on the screen. Say from 10 to 240. You want to insert a line between 30 and 40. No, I think I'll take a better one. You'll have a list from 610 to 700, or, my mental calculation is a little ... that is why I am a mathematician, I hate numbers. As a child when I found out that you didn't have to use numbers, that you could use letters instead, that was it!

OK, you have here some numbers from six hundred something to six hundred something else. You want to insert a line. You know then, if you have to use the line number. Very easy. Instead of saying 615, you write 15 for example. You introduce a wrong number. Later you cannot find that darn line. Finally you see it. What is doing there. It is between 10 and 20. It should be between 610 and 620. What you should be able to do, with or without line numbers, is to move your

cursor up to 620, press CTRL I - the text opens up, and you insert the line. So you get what you see. That's the whole idea. Also, if you put the cursor at 601, press CTRL D, and zip, the line disappears. This is what I am after. This is what I am trying to get the standards committee to accept at the next meeting. I know I'll have a hard time doing it because I know it is very difficult.

I got into an enormous lot of fights and arguments when I came out with the language.

That's another thing I might have told you. I got into an enormous lot of fights and arguments when I came out with the language. In fact, don't expect that people are happy and grateful, for your coming up with this. A lot of people would say: *Why what's this about, we are perfectly OK with what we have.*

There were articles in papers written against me, as though I was some new unknown form of Communist in disguise. It took me some years to understand why people can get mad at you when you come up with new ideas.

A programming language is not just a tool. It is something different and something more.

A programming language is not just a tool. It is something different and something more. In fact, what I was saying to some of these people ... or how I was interpreted was: *You are telling us that we are using bad language.* And that is certainly not a nice thing to tell people. Really! It is! Now I am much more careful not to be too tough. You think the saying comparing BASIC and the Devil was bad enough. Wow. You should have heard the arguments we used in Danish. They are absolutely untranslatable. They never should be anyway.

I hope I answered the question about the full screen.

What you are really after is a full screen editor. The line numbers are neither here nor there.

Well OK. In a 40 column screen it would give you 5 characters more. And sometimes it may prevent the wrap around. But that is minor. And whether the line numbers are going to be there or not is also of a minor concern. The important thing is that you have a full screen editor. You get what you see.

Referring to that problem, not only not being able to find the line that you were trying to insert when you mistyped it. About three days ago, I had one of those. I could not find it. Nothing at all worked. What I did was I double struck one of the keys and it typed in place of a line I had already written and everything went to pot.

Yes, that's right. If you could open up the text, you would know exactly what you were doing. You would see which line was pushed in order to make room for the next.

Well, I've seen a lot of this with my students. Coming up with a program and saying, well I had a program here 15 minutes ago, but ...

I remember, by the way, one experience we had with BASIC programs, and this is a true story again. Long John was the students name. He was almost 2 meters. He was a favorite with basketball. He was also a good programmer, but he came up one day with this, not a very big program. He asked me, in Danish of course, in a fair translation, he asked me the following question: *Borge, could you explain to me why this program works?*

Well, John. We sat down and we found out that three lines were never executed. He jumped around them. We sorted things out ... that was in the old bad BASIC days. He was only one out of a series of similar experiences. Now, if I should put a ... I could hardly use the good English work BASIC any more, but a basic philosophy, it would be the following: **Computers should be as easy to use for human beings as possible.**

We should be controlling computers, not the other way around. This was what was behind us all the time when we made COMAL.

We should be controlling computers, not the other way around. This was what was behind us all the time when we made this [COMAL]. Benedict Lofsted, the first one ... he was not in the second project because he went to Greenland, and that's a long ways from Denmark to Greenland. So, I had to the second project all alone. But it was not that

difficult, because it appeared that the first definition was so well designed that we could enhance it, extend it, without violating the original idea.

COMAL can take extensions without the original concept being ruined. Maybe that is why you like the language.

This is one of the other things, I hope you agree, that it still appears that COMAL can take extensions without the original concept being ruined. Maybe that is why you like the language in fact ... is the regularity in it ... or as a computer scientist would say, the orthagonality of it. I tried to learn that word.

One problem I have as my program gets growing to some length, that you start forgetting procedure names and things like that. Is it possible to have a window or something which could list the procedures that you have going at this point?

Yes. What you are asking is a list of the symbol table, of the variable names and the procedure names, probably with some indication of where they were to be found. We have that in the new CP/M COMAL. We have a LISTSYM or LISTPROC, I forget what we called it. Note that this has nothing to do with the language. This is the environment. And that is what is what we are after now. To make it even more friendly.

Is there a possibility of having certain procedures built in? For example, rounding up or that sort of situation?

Yes, it should be. That is what they are doing with the machine codes, the packages. In fact, you can build in anything. You can make yourself a commercial package.

Also, like COMAL 2.0 has built in graphics, if you want to draw a circle, you just use the keyword circle and its parameters. In 0.14 you have to have a procedure.

In the Mytech [Alder] version ... I hope it will eventually be a good version. You can even write the code [for packages] in the language C. [AmigaCOMAL also allows this] And C is much easier to use than machine code, in case you don't know it. And then use them as COMAL packages. That is to say, that all the procedures you write in machine code can be invoked or called as if they were normal user defined procedures.

I wanted to tell you a couple days ago, but I never got around to it. There is an environment, called artificial intelligence, developed at the University of Suffax, called PopWalk. It works in a screen oriented environment, providing you access to LISP, PROLOG, and POP11... that's the three languages. What I thought was extremely powerful, is that you could link to it subroutines, procedures, whatever you want to call them, in a host of other languages, including Pascal, Fortran, C. That strikes me as being extremely powerful. Each of these languages have some particular advantage, for some particular applications. Maybe it's the language you happen to know, that's already an advantage. Otherwise the language may be well suited to doing a particular task. So, I really try to encourage the idea that COMAL should be able to access routines compiled in a wide variety of Languages.

By the way, Seymour Papert (of Logo fame) took a look at COMAL, and he liked it!

I don't know whether that could be done, or how feasible it is, but it would be very practical to have it [in 1989 AmigaCOMAL and IBM PC COMAL 3.0 now allow this]. Of course you should be aware, that a language cannot be totally general. There are applications where even COMAL wouldn't cope with. We had a discussion with Seymour Papert, the man who made LOGO. He put out his position, quite clearly that, ... by the way, he took a look at COMAL, and he liked it! ... and he expressed his opinion that even children should try to learn more than one programming language. It will be more like programming concepts that they will be learning about. And it appears also that children are learning these things extremely fast. The robots, which I've brought one or two samples with me, you can look at them later if you like, were programmed by 11 year old children. We had no problems. The teacher had problems. The children had not.

Well, it seems, from what we've learned today of programming languages, that there are really three classes of languages. Each of them fulfill a role that the other ones cannot.

Yes. My guess is that in the future there will only be two. I think that the LISP languages will be taken over by the Logic, the PROLOG type. I think so. The three classes you think of are

probably the algorithmic languages, the LISP kind of languages, and the logic, PROLOG, kind of languages. COMAL is of course, an algorithmic language, it's in the name. LISP languages would be LISP or LOGO, LOGO is a LISP language. The third one would be PROLOG, a language that is mentioned very often, especially when you speak about 5th generation computers, and the competition between the United States and Japan. Who will be there first with this super thing. But I think that eventually you will see that there will be two. The algorithmic languages, they will never disappear. In robotics, they will still perform. Just imagine ADA. ADA is an algorithmic language, which by the way, COMAL would be a very good kindergarten ADA. It could be an introduction to ADA. I really think that. Pascal is an algorithmic language, Algol, etc. Whereas PROLOG,... and I think it is with PROLOG ... PROLOG has much the same relation to LISP, as Pascal would have to FORTRAN.

Well, the GOTO... Of course, in COMAL there are of course lots of GOTO, but they are hidden. The interpreter is doing them for you. IF, ELSE, ENDIF ... from IF you can jump to ELSE if the condition is not met. This is of course a GOTO, but it is hidden. Because it is being expressed in a way closer to the human language, or the human way of thinking. And the same way I guess that PROLOG would be able to handle lists in a way that would do the same to the list structures that the algorithmic structured languages have done to GOTO. That's my impression. But, OK. I'm sure of one thing. The algorithmic languages are there to stay.

Fortran even.

COMAL is first of all for people who are not professional ... to make it possible for people to program computers even if they were not programming people.

Yes, I think so. But, OK. COMAL is first of all for people who are not professional. I think I wrote in the handbook, to make it possible for people to program computers even if they were not programming people, or something like that. And it appears that it can also be used for some semi-professional, maybe even ...

[interrupts] Paradoxically, I think that the professionals are the fastest to appreciate it.

So much the better.

Of course, the trick is to get their attention. Once they see it, a little exposure to it, they immediately see that it is just the thing.

They have to see it to believe it. They must have a chance to see what COMAL is all about, to see a few examples, a few real programs.

This is probably the most important point you are indicating there. I heard one tonight who tried to introduce COMAL to some teachers somewhere. They have to see it to believe it. They must have a chance to see what COMAL is all about, to see a few examples, a few real programs, before they use it. You can give long talks, even very delicately formulated talks, about it. But there is nothing like a short example. You will see that, if you want to tell people about COMAL, you will have to show them a program in COMAL, and let them see the syntax, the error messages they get immediately, and the indentation that shows them... Let them look at an IF .. THEN .. ELSE, and REPEAT .. UNTIL, simple things, procedures can come later. As soon as they have seen that, they will know what it is about.

Now, I am running out of words. I can feel it. But don't hesitate to ask questions if you like.

Being one who has almost next to no experience in COMAL programming, I was interested if you could elaborate on those three levels of programming language that you just mentioned.

I could, but I am afraid it would be too technical. Do you know about LOGO and PROLOG. I would gladly do it afterwards. It is quite technical. The only thing I can say is, COMAL is a very typical algorithmic language. And if you want to know about a LISP language, the easiest way would be to look into LOGO, not at the turtle graphics! Because that has really nothing to do with LOGO. You can do turtle graphics in any language. You see it in USD Pascal for one. That was a real sensation. That he [Kenneth Bowles] used turtle graphics to train university students. And probably what made USD Pascal so popular.

You'll have to look into the data structures. LOGO is not using arrays like COMAL is. DIM a(10) or

whatever it would be. LOGO is using so called lists, which are more or less what mathematicians would call sets. PROLOG is totally different. It is the language that uses relations. It is more or less abstract algebra.

I have a simple question. Could you relate all the computers that you know that run COMAL now. We'll give you ten minutes.

Oh no. That would be boring.

No. I understand that there are some foreign computers. I would just like to know.

Well, in Denmark we have the old Z80 based microcomputers, which would look very much like Cromemco or IMSAI or whatever. In fact, Comet would look very much like the IMSAI, although the IMSAI used 8080 as far as I know. Then there are new ones coming up now, based on the 8086. Piccolo, and another one called Partner. More or less PC clones. And the Swedish Compis, which is the one Mytech [Alder] COMAL could be used on, also a 80186 based computer. Then the English Amstrad is one you may not know about. Apple has a COMAL in fact, Metanic COMAL. You have never seen that. It is not very good, I would not recommend it. A new one is being implemented right now. But only if the Apple has a Z80 card (CP/M). Then there will be PC COMAL of course. And the VAX computer. And Wicat. I don't know much about that, only that there is a COMAL for it. And Mytech [Alder] claim that they have one for the Mac. They call it MacCOMAL. Whether that is true or not, I don't know, I never saw it. [it is true ... we have seen the beta test of it working]. And I hate to speak about things I have not seen. Because in this branch, rumors ...

Have you considered introducing a help facility in the COMAL environment.

I hope that COMAL will go on developing. But you should expect it to be more in the environment.

Yes, Mytech [Alder] has a HELP. And the new Z80 version has a HELP. In fact the first test versions were running right before I left Denmark. This will be another thing that will be taken up by the Standardization Group. It is more or less the environment again. And it would be very useful I agree. I hope that COMAL will go on developing. But you should expect it to be more in the environment. The language as such I think is probably more or less finished. The definition is closed. There will not be added many more facilities. There are a few things, such as functions. You know that in 0.14 a function can return integers or reals, but not strings. Whereas in 2.0 functions can return strings. In Mytech [Alder] a function can return vectors, which might be an improvement in the language itself. Otherwise it would be to make it more and more user friendly.

What is the Mytech [Alder] version based on?

It is written in C. This means it can run on any computer that has a C compiler. Probably it would be very easy to do, if they haven't got one already, a 68000 version. We are trying to persuade them to put it on the Amiga, whenever that appears.

I've seen advertisements for PROMAL. Is that related to COMAL at all?

Not at all. I saw a German review of it. In Germany COMAL is very well know and is used extensively in many schools. The reviewer started to point out explicitly that PROMAL has nothing to do with COMAL.

Coming back to a point raised earlier. How about the possibility of compiling functions and procedures written in COMAL; compiling them directly into machine code. Once it works, it works forever, and then it runs a lot faster.

That would be a thing that probably anyone would like to have, a COMAL compiler. You would have something like the perfect system then.

That would of course be a thing that probably anyone would like to have, a COMAL compiler. You would have something like the perfect system then. The interpreter to develope interactively the program, to test it. As soon as the program runs as it should, you would take the program and compile it. It should be very easy to do, and I hope that someone will do it.

Let's let Borge get some rest.